

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

As demonstrated in the chart below, The Hisense Respondents. (“Hisense”) directly and indirectly infringes at least claim 1 of US 9,036,701 (the “’701 Patent”). Hisense directly infringes, contributes to the infringement of, and/or induces infringement of the ’701 Patent by making, using, selling, offering for sale, and/or importing into the United States the Accused Products that are covered by one or more claims of the ’701 Patent. The Accused Products are devices that decode H.265-compliant video . For example, the Hisense 43A7N is a representative product for other Hisense devices that decode H.265-compliant video.

The Hisense 43A7N contains at least one video decoder that helps decode H.265-compliant video.¹ While evidence from the Hisense 43A7N is specifically charted herein, the evidence and contentions charted herein apply equally to the other Hisense Accused Products that decode H.265-compliant video.

No part of this exemplary chart construes, or is intended to construe, the specification, file history, or claims of the ’701 Patent. Moreover, this exemplary chart does not limit, and is not intended to limit, Nokia’s infringement positions or contentions.

The following infringement chart includes exemplary citations to ITU-T Rec. H.265 (12/2016) High efficiency video coding (available at <https://www.itu.int/rec/T-REC-H.265-201612-S/en>) (the “H.265 Standard”). The cited functionality has been included in editions of the H.265 Standard since April 2013 and remains in current editions of the H.265 Standard. Any Hisense device that includes a decoder that practices the functionality in any of these editions of the H.265 Standard (“H.265 Decoder”) practices the claims of the ’701 Patent. Thus, the Accused Products each practice the H.265 Standard and are covered by claims of the ’701 Patent.

Nokia contends each of the following limitations is met literally, and, to the extent a limitation is not met literally, it is met under the doctrine of equivalents.²

¹ See, e.g., Hisense 43A7N User Manual available at <https://www.hisense-usa.com/televisions/hisense-43-class-a7-series-4k-wide-color-gamut-google-tv-43a7n>.

² This claim chart is based on the information currently available to Nokia and is intended to be exemplary in nature. Nokia reserves all rights to update and elaborate its infringement positions, including as Nokia obtains additional information during the course of discovery.

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

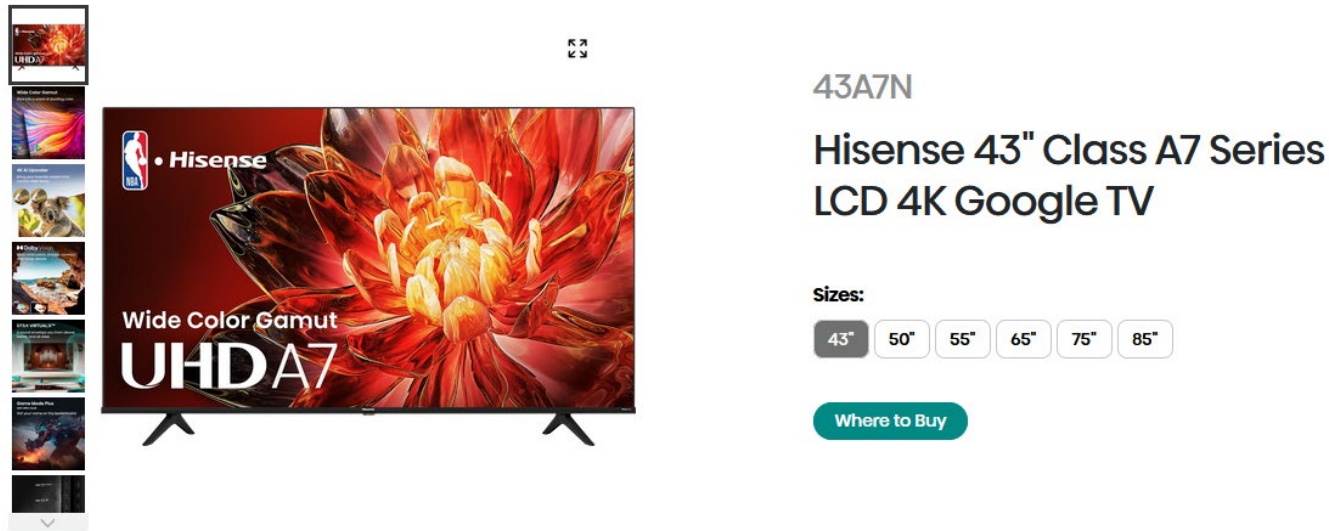
U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
<p>1. [Pre] A method comprising:</p>	<p>Each of the Accused Products, such as the Hisense 43A7N, performs a method comprising the limitations below.</p> <p>For example, and without limitation, the Hisense 43A7N uses hardware-accelerated video decoding and includes Mediatek MT9602 Processor.</p> <div data-bbox="613 506 1934 1034">  </div> <p>https://www.hisense-usa.com/televisions/hisense-43-class-a7-series-4k-wide-color-gamut-google-tv-43a7n (last accessed March 29, 2025).</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS


U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<div data-bbox="632 462 1203 824">  </div> <div data-bbox="1234 358 1795 654"> <p>Display: 42.5 in, VA, Direct LED, 3840 x 2160 pixels Viewing angles (H/V): 178 ° / 178 ° Brightness: 275 cd/m² Static contrast: 4000 : 1 Refresh rate: 50 Hz / 60 Hz Frame interpolation: 120 MR (Motion Rate) TV tuner: Analog (NTSC/PAL/SECAM), ATSC, Clear QAM SoC: MediaTek MT9602 CPU: ARM Cortex-A53, 1500 MHz, Cores: 4 Dimensions: 963 x 560 x 74 mm Weight: 6.8 kg</p> </div> <div data-bbox="1241 678 1740 727"> <div>Add to compare</div> <div>Suggest an edit</div> </div> <div data-bbox="613 943 1365 979"> https://www.displayspecifications.com/en/model/234a3f3b </div>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS			
	Video Format			
	Container	Video Codec	File Extension Name	Resolution and Frame Rate
	MPEG program stream	MPEG1/2	.DAT, .VOB, .MPG, .MPEG	1920 x 1080 @ 60fps
		MPEG4		
		H.264		3840 x 2160 @ 60fps
	MPEG transport stream	HEVC/H.265	.ts, .trp, .tp	3840 x 2160 @ 60fps
		MPEG4		1920 x 1080 @ 60fps
		H.264		3840 x 2160 @ 60fps
		VC1		1920 x 1080 @ 60fps

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS			
	Container	Video Codec	File Extension Name	Resolution and Frame Rate
		MPEG1/2		1920 x 1080 @ 60fps
		AVS		
		AVS+		
		AVS2		3840 x 2160 @ 60fps
	MP4	VP8	.mp4, .mov	1920 x 1080 @ 60fps
		AV1		3840 x 2160 @ 60fps
		HEVC/H.265		
		MPEG1/2		1920 x 1080 @ 60fps
		MPEG4		
		H.263		
		H.264		3840 x 2160 @ 60fps
		WMV3		1920 x 1080 @ 60fps
		VC1		
	Motion JPEG	1920 x 1080 @ 30fps		
Source: Hisense 43A7N User Manual, at 45-46. Downloaded from https://www.hisense-usa.com/televisions/hisense-43-class-a7-series-4k-wide-color-gamut-google-tv-43a7n (last accessed March 29, 2025).				
For example, an Hisense 43A7N was used to playback an H.265-compliant video.				

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

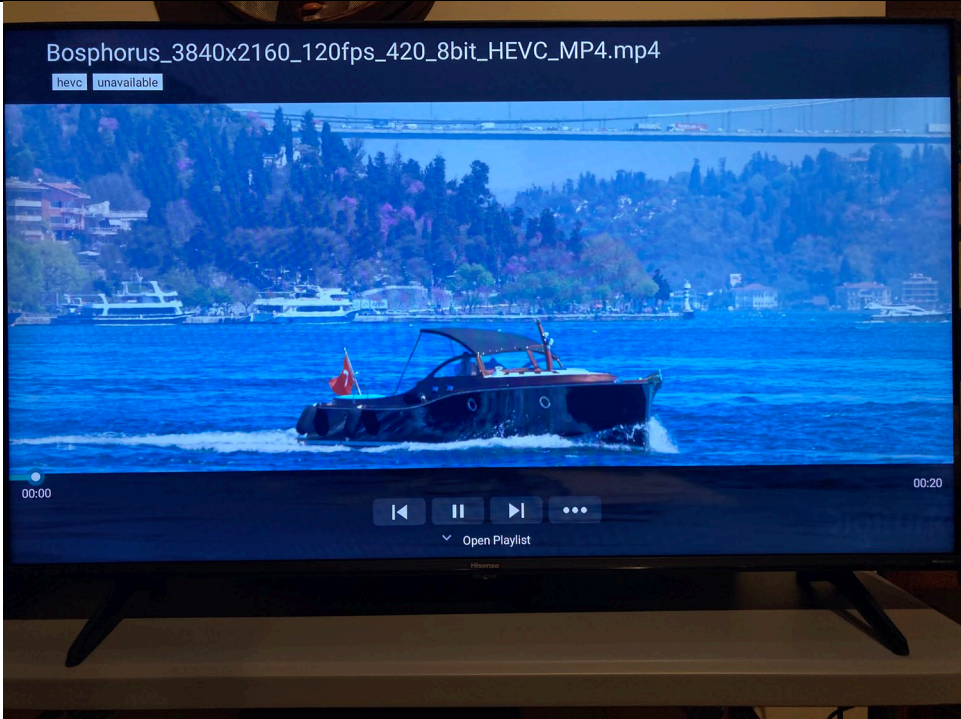
U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<div data-bbox="611 280 1566 995">  </div> <p data-bbox="611 998 1518 1032">Source: Picture of H.265-complaint video playback on Hisense 43A7N</p> <p data-bbox="611 1073 1978 1143">For example, and without limitation, the H.265 Standard specifies the following regarding the decoding process. Each of the Accused Products performs a method comprising the limitations below.</p> <p data-bbox="686 1219 861 1252">3 Definitions</p> <p data-bbox="686 1271 1944 1305">For the purposes of this Recommendation International Standard, the following definitions apply.</p> <p data-bbox="686 1336 728 1357">...</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<p>3.12 bitstream: A sequence of bits, . . . , that forms the representation of <i>coded pictures</i> and associated data forming one or more coded video sequences (<i>CVSs</i>).</p> <p>. . .</p> <p>3.25 coded picture: A <i>coded representation</i> of a picture . . .</p> <p>. . .</p> <p>3.44 decoding process: The process specified in this Specification that reads a <i>bitstream</i> and derives <i>decoded pictures</i> from it.</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 4 – 7.</p>
<p>[a] determining a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream;</p>	<p>Each of the Accused Products, such as the Hisense 43A7N, performs a method comprising determining a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream.</p> <p>For example, and without limitation, the H.265 Standard specifies the following regarding the decoding process. Each of the Accused Products performs a method comprising determining a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream.</p> <p>The following specifications provide further evidence of how each of the Accused Products operates:</p> <p>3 Definitions</p> <p>For the purposes of this Recommendation International Standard, the following definitions apply.</p> <p>. . .</p> <p>3.153 syntax element: <i>An element of data represented in the bitstream.</i></p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS				
	<p>...</p> <p>3.166 transform coefficient: A Scalar quantity, considered to be in a frequency domain, that is associated with a particular one-dimensional or two-dimensional <i>frequency index</i> in a <i>transform</i> in the <i>decoding process</i>.</p> <p>3.167 transform coefficient level: An integer quantity representing the value associated with a particular two-dimensional frequency index in the decoding process prior to scaling for computation of a transform coefficient value.</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 12.</p> <p>7.3.8.11 Residual coding syntax</p> <table border="1" data-bbox="674 820 1913 930"> <tr> <td data-bbox="674 820 1755 862">residual_coding(x0, y0, log2TrafoSize, cIdx) {</td><td data-bbox="1755 820 1913 862">Descriptor</td></tr> <tr> <td data-bbox="674 862 1755 930">if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize <= Log2MaxTransformSkipSize))</td><td data-bbox="1755 862 1913 930"></td></tr> </table> <p>...</p>	residual_coding(x0, y0, log2TrafoSize, cIdx) {	Descriptor	if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize <= Log2MaxTransformSkipSize))	
residual_coding(x0, y0, log2TrafoSize, cIdx) {	Descriptor				
if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize <= Log2MaxTransformSkipSize))					

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS	
	<pre> numGreater1Flag = 0 lastGreater1ScanPos = -1 for(n = 15; n >= 0; n--) { xC = (xS << 2) + ScanOrder[2][scanIdx][n][0] yC = (yS << 2) + ScanOrder[2][scanIdx][n][1] if(sig_coeff_flag[xC][yC]) { if(numGreater1Flag < 8) { coeff_abs_level_greater1_flag[n] numGreater1Flag++ if(coeff_abs_level_greater1_flag[n] && lastGreater1ScanPos == -1) lastGreater1ScanPos = n else if(coeff_abs_level_greater1_flag[n]) escapeDataPresent = 1 } else </pre>	<p>ae(v)</p>
<p>[b] categorizing a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of</p>	<p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 58-59.</p> <p>Each of the Accused Products, such as the Hisense 43A7N, performs a method comprising categorizing a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of occurrence threshold are categorized into the second category and syntax elements which occur less than the frequency of occurrence are categorized into the first category.</p> <p>For example, and without limitation, the H.265 Standard specifies the following regarding the decoding process. Each of the Accused Products performs a method comprising</p>	

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
<p>occurrence threshold are categorized into the second category and syntax elements which occur less than the frequency of occurrence are categorized into the first category;</p>	<p>categorizing a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of occurrence threshold are categorized into the second category and syntax elements which occur less than the frequency of occurrence are categorized into the first category.</p> <p>The following specifications provide further evidence of how each of the Accused Products operates:</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS																																				
	<p>7.3.8.11 Residual coding syntax</p> <table border="1"> <thead> <tr> <th data-bbox="669 370 1772 406">residual_coding(x0, y0, log2TrafoSize, cIdx) {</th><th data-bbox="1772 370 1934 406">Descriptor</th></tr> </thead> <tbody> <tr> <td data-bbox="669 406 1772 483">if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize <= Log2MaxTransformSkipSize))</td><td data-bbox="1772 406 1934 483"></td></tr> <tr> <td data-bbox="669 483 1772 526">transform_skip_flag[x0][y0][cIdx]</td><td data-bbox="1772 483 1934 526">ae(v)</td></tr> <tr> <td data-bbox="669 526 1772 604">if(CuPredMode[x0][y0] == MODE_INTER && explicit_rdpem_enabled_flag && (transform_skip_flag[x0][y0][cIdx] cu_transquant_bypass_flag)) {</td><td data-bbox="1772 526 1934 604"></td></tr> <tr> <td data-bbox="669 604 1772 646">explicit_rdpem_flag[x0][y0][cIdx]</td><td data-bbox="1772 604 1934 646">ae(v)</td></tr> <tr> <td data-bbox="669 646 1772 688">if(explicit_rdpem_flag[x0][y0][cIdx])</td><td data-bbox="1772 646 1934 688"></td></tr> <tr> <td data-bbox="669 688 1772 730">explicit_rdpem_dir_flag[x0][y0][cIdx]</td><td data-bbox="1772 688 1934 730">ae(v)</td></tr> <tr> <td data-bbox="669 730 1772 773">}</td><td data-bbox="1772 730 1934 773"></td></tr> <tr> <td data-bbox="669 773 1772 815">last_sig_coeff_x_prefix</td><td data-bbox="1772 773 1934 815">ae(v)</td></tr> <tr> <td data-bbox="669 815 1772 857">last_sig_coeff_y_prefix</td><td data-bbox="1772 815 1934 857">ae(v)</td></tr> <tr> <td data-bbox="669 857 1772 899">if(last_sig_coeff_x_prefix > 3)</td><td data-bbox="1772 857 1934 899"></td></tr> <tr> <td data-bbox="669 899 1772 941">last_sig_coeff_x_suffix</td><td data-bbox="1772 899 1934 941">ae(v)</td></tr> <tr> <td data-bbox="669 941 1772 984">if(last_sig_coeff_y_prefix > 3)</td><td data-bbox="1772 941 1934 984"></td></tr> <tr> <td data-bbox="669 984 1772 1026">last_sig_coeff_y_suffix</td><td data-bbox="1772 984 1934 1026">ae(v)</td></tr> <tr> <td data-bbox="669 1026 1772 1068">lastScanPos = 16</td><td data-bbox="1772 1026 1934 1068"></td></tr> <tr> <td data-bbox="669 1068 1772 1110">lastSubBlock = (1 << (log2TrafoSize - 2)) * (1 << (log2TrafoSize - 2)) - 1</td><td data-bbox="1772 1068 1934 1110"></td></tr> <tr> <td data-bbox="669 1110 1772 1153">do {</td><td data-bbox="1772 1110 1934 1153"></td></tr> <tr> <td data-bbox="669 1153 1772 1195">if(lastScanPos == 0) {</td><td data-bbox="1772 1153 1934 1195"></td></tr> </tbody> </table>	residual_coding(x0, y0, log2TrafoSize, cIdx) {	Descriptor	if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize <= Log2MaxTransformSkipSize))		transform_skip_flag[x0][y0][cIdx]	ae(v)	if(CuPredMode[x0][y0] == MODE_INTER && explicit_rdpem_enabled_flag && (transform_skip_flag[x0][y0][cIdx] cu_transquant_bypass_flag)) {		explicit_rdpem_flag[x0][y0][cIdx]	ae(v)	if(explicit_rdpem_flag[x0][y0][cIdx])		explicit_rdpem_dir_flag[x0][y0][cIdx]	ae(v)	}		last_sig_coeff_x_prefix	ae(v)	last_sig_coeff_y_prefix	ae(v)	if(last_sig_coeff_x_prefix > 3)		last_sig_coeff_x_suffix	ae(v)	if(last_sig_coeff_y_prefix > 3)		last_sig_coeff_y_suffix	ae(v)	lastScanPos = 16		lastSubBlock = (1 << (log2TrafoSize - 2)) * (1 << (log2TrafoSize - 2)) - 1		do {		if(lastScanPos == 0) {	
residual_coding(x0, y0, log2TrafoSize, cIdx) {	Descriptor																																				
if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize <= Log2MaxTransformSkipSize))																																					
transform_skip_flag[x0][y0][cIdx]	ae(v)																																				
if(CuPredMode[x0][y0] == MODE_INTER && explicit_rdpem_enabled_flag && (transform_skip_flag[x0][y0][cIdx] cu_transquant_bypass_flag)) {																																					
explicit_rdpem_flag[x0][y0][cIdx]	ae(v)																																				
if(explicit_rdpem_flag[x0][y0][cIdx])																																					
explicit_rdpem_dir_flag[x0][y0][cIdx]	ae(v)																																				
}																																					
last_sig_coeff_x_prefix	ae(v)																																				
last_sig_coeff_y_prefix	ae(v)																																				
if(last_sig_coeff_x_prefix > 3)																																					
last_sig_coeff_x_suffix	ae(v)																																				
if(last_sig_coeff_y_prefix > 3)																																					
last_sig_coeff_y_suffix	ae(v)																																				
lastScanPos = 16																																					
lastSubBlock = (1 << (log2TrafoSize - 2)) * (1 << (log2TrafoSize - 2)) - 1																																					
do {																																					
if(lastScanPos == 0) {																																					

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS																																																				
	<table border="1"> <tr><td>lastScanPos = 16</td><td></td></tr> <tr><td>lastSubBlock--</td><td></td></tr> <tr><td>}</td><td></td></tr> <tr><td>lastScanPos--</td><td></td></tr> <tr><td>xS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][0]</td><td></td></tr> <tr><td>yS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][1]</td><td></td></tr> <tr><td>xC = (xS << 2) + ScanOrder[2][scanIdx][lastScanPos][0]</td><td></td></tr> <tr><td>yC = (yS << 2) + ScanOrder[2][scanIdx][lastScanPos][1]</td><td></td></tr> <tr><td>} while((xC != LastSignificantCoeffX) (yC != LastSignificantCoeffY))</td><td></td></tr> <tr><td>for(i = lastSubBlock; i >= 0; i--) {</td><td></td></tr> <tr><td> xS = ScanOrder[log2TrafoSize - 2][scanIdx][i][0]</td><td></td></tr> <tr><td> yS = ScanOrder[log2TrafoSize - 2][scanIdx][i][1]</td><td></td></tr> <tr><td> escapeDataPresent = 0</td><td></td></tr> <tr><td> inferSbDcSigCoeffFlag = 0</td><td></td></tr> <tr><td> if((i < lastSubBlock) && (i > 0)) {</td><td></td></tr> <tr><td> coded_sub_block_flag[xS][yS]</td><td>ae(v)</td></tr> <tr><td> inferSbDcSigCoeffFlag = 1</td><td></td></tr> <tr><td> }</td><td></td></tr> <tr><td> for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {</td><td></td></tr> <tr><td> xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]</td><td></td></tr> <tr><td> yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]</td><td></td></tr> <tr><td> if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {</td><td></td></tr> <tr><td> sig_coeff_flag[xC][yC]</td><td>ae(v)</td></tr> <tr><td> if(sig_coeff_flag[xC][yC])</td><td></td></tr> <tr><td> inferSbDcSigCoeffFlag = 0</td><td></td></tr> <tr><td> }</td><td></td></tr> </table>	lastScanPos = 16		lastSubBlock--		}		lastScanPos--		xS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][0]		yS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][1]		xC = (xS << 2) + ScanOrder[2][scanIdx][lastScanPos][0]		yC = (yS << 2) + ScanOrder[2][scanIdx][lastScanPos][1]		} while((xC != LastSignificantCoeffX) (yC != LastSignificantCoeffY))		for(i = lastSubBlock; i >= 0; i--) {		xS = ScanOrder[log2TrafoSize - 2][scanIdx][i][0]		yS = ScanOrder[log2TrafoSize - 2][scanIdx][i][1]		escapeDataPresent = 0		inferSbDcSigCoeffFlag = 0		if((i < lastSubBlock) && (i > 0)) {		coded_sub_block_flag[xS][yS]	ae(v)	inferSbDcSigCoeffFlag = 1		}		for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {		xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]		yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]		if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {		sig_coeff_flag[xC][yC]	ae(v)	if(sig_coeff_flag[xC][yC])		inferSbDcSigCoeffFlag = 0		}	
lastScanPos = 16																																																					
lastSubBlock--																																																					
}																																																					
lastScanPos--																																																					
xS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][0]																																																					
yS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][1]																																																					
xC = (xS << 2) + ScanOrder[2][scanIdx][lastScanPos][0]																																																					
yC = (yS << 2) + ScanOrder[2][scanIdx][lastScanPos][1]																																																					
} while((xC != LastSignificantCoeffX) (yC != LastSignificantCoeffY))																																																					
for(i = lastSubBlock; i >= 0; i--) {																																																					
xS = ScanOrder[log2TrafoSize - 2][scanIdx][i][0]																																																					
yS = ScanOrder[log2TrafoSize - 2][scanIdx][i][1]																																																					
escapeDataPresent = 0																																																					
inferSbDcSigCoeffFlag = 0																																																					
if((i < lastSubBlock) && (i > 0)) {																																																					
coded_sub_block_flag[xS][yS]	ae(v)																																																				
inferSbDcSigCoeffFlag = 1																																																					
}																																																					
for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {																																																					
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]																																																					
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]																																																					
if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {																																																					
sig_coeff_flag[xC][yC]	ae(v)																																																				
if(sig_coeff_flag[xC][yC])																																																					
inferSbDcSigCoeffFlag = 0																																																					
}																																																					

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS																																				
	<table border="1"> <tr><td data-bbox="676 440 1770 483">}</td><td data-bbox="1770 440 1927 483"></td></tr> <tr><td data-bbox="676 483 1770 527">firstSigScanPos = 16</td><td data-bbox="1770 483 1927 527"></td></tr> <tr><td data-bbox="676 527 1770 571">lastSigScanPos = -1</td><td data-bbox="1770 527 1927 571"></td></tr> <tr><td data-bbox="676 571 1770 615">numGreater1Flag = 0</td><td data-bbox="1770 571 1927 615"></td></tr> <tr><td data-bbox="676 615 1770 659">lastGreater1ScanPos = -1</td><td data-bbox="1770 615 1927 659"></td></tr> <tr><td data-bbox="676 659 1770 703">for(n = 15; n >= 0; n--) {</td><td data-bbox="1770 659 1927 703"></td></tr> <tr><td data-bbox="676 703 1770 747">xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]</td><td data-bbox="1770 703 1927 747"></td></tr> <tr><td data-bbox="676 747 1770 790">yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]</td><td data-bbox="1770 747 1927 790"></td></tr> <tr><td data-bbox="676 790 1770 834">if(sig_coeff_flag[xC][yC]) {</td><td data-bbox="1770 790 1927 834"></td></tr> <tr><td data-bbox="676 834 1770 878">if(numGreater1Flag < 8) {</td><td data-bbox="1770 834 1927 878"></td></tr> <tr><td data-bbox="676 878 1770 922">coeff_abs_level_greater1_flag[n]</td><td data-bbox="1770 878 1927 922">ae(v)</td></tr> <tr><td data-bbox="676 922 1770 966">numGreater1Flag++</td><td data-bbox="1770 922 1927 966"></td></tr> <tr><td data-bbox="676 966 1770 1010">if(coeff_abs_level_greater1_flag[n] && lastGreater1ScanPos == -1)</td><td data-bbox="1770 966 1927 1010"></td></tr> <tr><td data-bbox="676 1010 1770 1053">lastGreater1ScanPos = n</td><td data-bbox="1770 1010 1927 1053"></td></tr> <tr><td data-bbox="676 1053 1770 1097">else if(coeff_abs_level_greater1_flag[n])</td><td data-bbox="1770 1053 1927 1097"></td></tr> <tr><td data-bbox="676 1097 1770 1141">escapeDataPresent = 1</td><td data-bbox="1770 1097 1927 1141"></td></tr> <tr><td data-bbox="676 1141 1770 1185">} else</td><td data-bbox="1770 1141 1927 1185"></td></tr> <tr><td data-bbox="676 1185 1770 1229">escapeDataPresent = 1</td><td data-bbox="1770 1185 1927 1229"></td></tr> </table>	}		firstSigScanPos = 16		lastSigScanPos = -1		numGreater1Flag = 0		lastGreater1ScanPos = -1		for(n = 15; n >= 0; n--) {		xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]		yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]		if(sig_coeff_flag[xC][yC]) {		if(numGreater1Flag < 8) {		coeff_abs_level_greater1_flag[n]	ae(v)	numGreater1Flag++		if(coeff_abs_level_greater1_flag[n] && lastGreater1ScanPos == -1)		lastGreater1ScanPos = n		else if(coeff_abs_level_greater1_flag[n])		escapeDataPresent = 1		} else		escapeDataPresent = 1	
}																																					
firstSigScanPos = 16																																					
lastSigScanPos = -1																																					
numGreater1Flag = 0																																					
lastGreater1ScanPos = -1																																					
for(n = 15; n >= 0; n--) {																																					
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]																																					
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]																																					
if(sig_coeff_flag[xC][yC]) {																																					
if(numGreater1Flag < 8) {																																					
coeff_abs_level_greater1_flag[n]	ae(v)																																				
numGreater1Flag++																																					
if(coeff_abs_level_greater1_flag[n] && lastGreater1ScanPos == -1)																																					
lastGreater1ScanPos = n																																					
else if(coeff_abs_level_greater1_flag[n])																																					
escapeDataPresent = 1																																					
} else																																					
escapeDataPresent = 1																																					

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS	
	if(lastSigScanPos == -1)	
	lastSigScanPos = n	
	firstSigScanPos = n	
	}	
	}	
	if(cu_transquant_bypass_flag (CuPredMode[x0][y0] == MODE_INTRA && implicit_rdpdm_enabled_flag && transform_skip_flag[x0][y0][cIdx] && (predModeIntra == 10 predModeIntra == 26)) explicit_rdpdm_flag[x0][y0][cIdx])	
	signHidden = 0	
	else	
	signHidden = lastSigScanPos - firstSigScanPos > 3	
	if(lastGreater1ScanPos != -1) {	
	coeff_abs_level_greater2_flag[lastGreater1ScanPos]	ae(v)
	if(coeff_abs_level_greater2_flag[lastGreater1ScanPos])	
	escapeDataPresent = 1	
	}	

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS		
		for(n = 15; n >= 0; n--) {	
		xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
		yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
		if(sig_coeff_flag[xC][yC] && (!sign_data_hiding_enabled_flag !signHidden (n != firstSigScanPos)))	
		coeff_sign_flag[n]	ae(v)
		}	

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<pre> numSigCoeff = 0 sumAbsLevel = 0 for(n = 15; n >= 0; n--) { xC = (xS << 2) + ScanOrder[2][scanIdx][n][0] yC = (yS << 2) + ScanOrder[2][scanIdx][n][1] if(sig_coeff_flag[xC][yC]) { baseLevel = 1 + coeff_abs_level_greater1_flag[n] + coeff_abs_level_greater2_flag[n] if(baseLevel == ((numSigCoeff < 8) ? (n == lastGreater1ScanPos) ? 3 : 2) : 1)) coeff_abs_level_remaining[n] TransCoeffLevel[x0][y0][cIdx][xC][yC] = (coeff_abs_level_remaining[n] + baseLevel) * (1 - 2 * coeff_sign_flag[n]) if(sign_data_hiding_enabled_flag && signHidden) { sumAbsLevel += (coeff_abs_level_remaining[n] + baseLevel) if((n == firstSigScanPos) && ((sumAbsLevel % 2) == 1)) TransCoeffLevel[x0][y0][cIdx][xC][yC] = -TransCoeffLevel[x0][y0][cIdx][xC][yC] } numSigCoeff++ } } } } </pre>
	<p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 58-60.</p> <p>As set out in the Residual Coding Syntax in section 7.3.8.11, an H.265 compliant decoder parses <i>sig_coeff_flags</i> from the bit stream according to the logic reproduced below.</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS																
	<table border="1" data-bbox="667 321 1961 665"> <tr> <td>for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {</td><td></td></tr> <tr> <td>xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]</td><td></td></tr> <tr> <td>yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]</td><td></td></tr> <tr> <td>if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {</td><td></td></tr> <tr> <td>sig_coeff_flag[xC][yC]</td><td>ae(v)</td></tr> <tr> <td>if(sig_coeff_flag[xC][yC])</td><td></td></tr> <tr> <td>inferSbDcSigCoeffFlag = 0</td><td></td></tr> <tr> <td>}</td><td></td></tr> </table> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 59.</p> <p>7.4.9.11 Residual coding semantics</p> <p>...</p> <p>sig_coeff_flag[xC][yC] specifies for the transform coefficient location (xC, yC) within the current transform block whether the corresponding transform coefficient level at the location (xC, yC) is non-zero as follows:</p> <ul style="list-style-type: none"> – If sig_coeff_flag[xC][yC] is equal to 0, the transform coefficient level at the location (xC, yC) is set equal to 0. – Otherwise (sig_coeff_flag[xC][yC] is equal to 1), the transform coefficient level at the location (xC, yC) has a non-zero value. <p>When sig_coeff_flag[xC][yC] is not present, it is inferred as follows:</p> <ul style="list-style-type: none"> – If (xC, yC) is the last significant location (LastSignificantCoeffX, LastSignificantCoeffY) in scan order or all of the following conditions are true, sig_coeff_flag[xC][yC] is inferred to be equal to 1: <ul style="list-style-type: none"> – (xC & 3, yC & 3) is equal to (0, 0). – inferSbDcSigCoeffFlag is equal to 1. 	for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {		xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]		yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]		if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {		sig_coeff_flag[xC][yC]	ae(v)	if(sig_coeff_flag[xC][yC])		inferSbDcSigCoeffFlag = 0		}	
for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {																	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]																	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]																	
if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {																	
sig_coeff_flag[xC][yC]	ae(v)																
if(sig_coeff_flag[xC][yC])																	
inferSbDcSigCoeffFlag = 0																	
}																	

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<ul style="list-style-type: none"> – coded_sub_block_flag[xS][yS] is equal to 1. – Otherwise, sig_coeff_flag[xC][yC] is inferred to be equal to 0. ... <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 111.</p> <p>9.3.4.2 Derivation process for ctxTable, ctxIdx and bypassFlag</p> <p>9.3.4.2.1 General</p> <p>Input to this process is the position of the current bin within the bin string, binIdx.</p> <p>Outputs of this process are ctxTable, ctxIdx and bypassFlag.</p> <p>The values of ctxTable, ctxIdx and bypassFlag are derived as follows based on the entries for binIdx of the corresponding syntax element in Table 9-48:</p> <ul style="list-style-type: none"> – If the entry in Table 9-48 is not equal to "bypass", "terminate" or "na", the values of binIdx are decoded by invoking the DecodeDecision process as specified in clause 9.3.4.3.2 and the following applies: <ul style="list-style-type: none"> – ctxTable is specified in Table 9-4. – The variable ctxInc is specified by the corresponding entry in Table 9-48 and when more than one value is listed in Table 9-48 for a binIdx, the assignment process for ctxInc for that binIdx is further specified in the clauses given in parenthesis. – The variable ctxIdxOffset is specified by the lowest value of ctxIdx in Table 9-4 depending on the current value of initType. – ctxIdx is set equal to the sum of ctxInc and ctxIdxOffset. – bypassFlag is set equal to 0. – Otherwise, if the entry in Table 9-48 is equal to "bypass", the values of binIdx are decoded by invoking the DecodeBypass process as specified in clause 9.3.4.3.4 and the following applies:

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<div><div><div><div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></</div></div></div></div></div>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<p>The following specifications provide further evidence of how each of the Accused Products operates:</p> <p>9.3.4.2 Derivation process for ctxTable, ctxIdx and bypassFlag</p> <p>9.3.4.2.1 General</p> <p>Input to this process is the position of the current bin within the bin string, binIdx.</p> <p>Outputs of this process are ctxTable, ctxIdx and bypassFlag.</p> <p>The values of ctxTable, ctxIdx and bypassFlag are derived as follows based on the entries for binIdx of the corresponding syntax element in Table 9-48:</p> <ul style="list-style-type: none"> – If the entry in Table 9-48 is not equal to "bypass", "terminate" or "na", the values of binIdx are decoded by invoking the DecodeDecision process as specified in clause 9.3.4.3.2 and the following applies: <ul style="list-style-type: none"> – ctxTable is specified in Table 9-4. – The variable ctxInc is specified by the corresponding entry in Table 9-48 and when more than one value is listed in Table 9-48 for a binIdx, the assignment process for ctxInc for that binIdx is further specified in the clauses given in parenthesis. – The variable ctxIdxOffset is specified by the lowest value of ctxIdx in Table 9-4 depending on the current value of initType. – ctxIdx is set equal to the sum of ctxInc and ctxIdxOffset. – bypassFlag is set equal to 0. ... <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 225.</p> <p>9.3.4.3 Arithmetic decoding process</p> <p>9.3.4.3.1 General</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<p>Inputs to this process are ctxTable, ctxIdx and bypassFlag, as derived in clause 9.3.4.2, and the state variables ivlCurrRange and ivlOffset of the arithmetic decoding engine.</p> <p>Output of this process is the value of the bin.</p> <p>Figure 9-5 illustrates the whole arithmetic decoding process for a single bin. For decoding the value of a bin, the context index table ctxTable and the ctxIdx are passed to the arithmetic decoding process DecodeBin(ctxTable, ctxIdx), which is specified as follows:</p> <ul style="list-style-type: none"> – If bypassFlag is equal to 1, DecodeBypass() as specified in clause 9.3.4.3.4 is invoked. – Otherwise, if bypassFlag is equal to 0, ctxTable is equal to 0 and ctxIdx is equal to 0, DecodeTerminate() as specified in clause 9.3.4.3.5 is invoked. – Otherwise (bypassFlag is equal to 0 and ctxTable is not equal to 0), DecodeDecision() as specified in clause 9.3.4.3.2 is invoked. <p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 232.</p> <p>9.3.4.2.6 Derivation process of ctxInc for the syntax element coeff_abs_level_greater1_flag</p> <p>Inputs to this process are the colour component index cIdx, the current sub-block scan index i and the current coefficient scan index n within the current sub-block.</p> <p>Output of this process is the variable ctxInc.</p> <p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p 231.</p> <p>9.3.4.2.7 Derivation process of ctxInc for the syntax element coeff_abs_level_greater2_flag</p> <p>Inputs to this process are the colour component index cIdx, the current sub-block scan index i and the current coefficient scan index n within the current sub-block.</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS																																																
	<p>Output of this process is the variable ctxInc.</p> <p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 232.</p> <p>The symbols that correspond to the first category of syntax elements and that have been subjected to a context update are the coeff_abs_level_greater1_flag, and coeff_abs_level_greater2_flag that are used to represent, at least in part, the <i>non-zero transform coefficient levels</i> in the first category, as shown in Table 9-48 and described in section 9.3.2.5 of the H.265 Standard reproduced below, in which the process for calculating the context for each of coeff_abs_level_greater1_flag and coeff_abs_level_greater2_flag is provided.</p> <p style="text-align: center;">Table 9-48 – Assignment of ctxInc to syntax elements with context coded bins</p> <table><tr><th rowspan="2">Syntax element</th><th colspan="6">binIdx</th></tr><tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>>= 5</th></tr><tr><td>...</td><td colspan="6"></td></tr><tr><td>sig_coeff_flag[][]</td><td>0..43 (clause 9.3.4.2.5)</td><td>na</td><td>na</td><td>na</td><td>na</td><td>na</td></tr><tr><td>coeff_abs_level_greater1_flag[]</td><td>0..23 (clause 9.3.4.2.6)</td><td>na</td><td>na</td><td>na</td><td>na</td><td>na</td></tr><tr><td>coeff_abs_level_greater2_flag[]</td><td>0..5 (clause 9.3.4.2.7)</td><td>na</td><td>na</td><td>na</td><td>na</td><td>na</td></tr><tr><td>coeff_abs_level_remaining[]</td><td>bypass</td><td>bypass</td><td>bypass</td><td>bypass</td><td>bypass</td><td>bypass</td></tr></table> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 225-227.</p> <p>9.3.2.5 Synchronization process for context variables, Rice parameter initialization states, and palette predictor variables</p>	Syntax element	binIdx						0	1	2	3	4	>= 5	...							sig_coeff_flag[][]	0..43 (clause 9.3.4.2.5)	na	na	na	na	na	coeff_abs_level_greater1_flag[]	0..23 (clause 9.3.4.2.6)	na	na	na	na	na	coeff_abs_level_greater2_flag[]	0..5 (clause 9.3.4.2.7)	na	na	na	na	na	coeff_abs_level_remaining[]	bypass	bypass	bypass	bypass	bypass	bypass
Syntax element	binIdx																																																
	0	1	2	3	4	>= 5																																											
...																																																	
sig_coeff_flag[][]	0..43 (clause 9.3.4.2.5)	na	na	na	na	na																																											
coeff_abs_level_greater1_flag[]	0..23 (clause 9.3.4.2.6)	na	na	na	na	na																																											
coeff_abs_level_greater2_flag[]	0..5 (clause 9.3.4.2.7)	na	na	na	na	na																																											
coeff_abs_level_remaining[]	bypass	bypass	bypass	bypass	bypass	bypass																																											

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<p>Inputs to this process are:</p> <ul style="list-style-type: none"> – The variables tableStateSync and tableMPSSync containing the values of the variables pStateIdx and valMps used in the storage process of context variables that are assigned to all syntax elements in clauses 7.3.8.1 through 7.3.8.12, except end_of_slice_segment_flag, end_of_subset_one_bit and pcm_flag. – The variable tableStatCoeffSync containing the values of the variables StatCoeff[k] used in the storage process of context variables and Rice parameter initialization states. – The variables PredictorPaletteSizeSync and tablePredictorPaletteEntriesSync containing the values used in the storage process of palette predictor variables. <p>Outputs of this process are:</p> <ul style="list-style-type: none"> – The initialized CABAC context variables indexed by ctxTable and ctxIdx. – The initialized Rice parameter initialization states StatCoeff indexed by k. – The palette predictor variables, PredictorPaletteSize and PredictorPaletteEntries. <p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 215.</p>
<p>[d] entropy coding symbols that correspond to the second category of syntax elements and that have bypassed context updating.</p>	<p>Each of the Accused Products, such as the Hisense 43A7N, performs a method comprising entropy coding symbols that correspond to the second category of syntax elements and that have bypassed context updating.</p> <p>For example, and without limitation, the H.265 Standard specifies the following regarding the decoding process. Each of the Accused Products performs a method comprising entropy coding symbols that correspond to the second category of syntax elements and that have bypassed context updating.</p> <p>The following specifications provide further evidence of how each of the Accused Products operates:</p> <p style="text-align: center;">9.3.4.2 Derivation process for ctxTable, ctxIdx and bypassFlag</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS																																																
	<p>9.3.4.2.1 General</p> <p>Input to this process is the position of the current bin within the bin string, binIdx.</p> <p>Outputs of this process are ctxTable, ctxIdx and bypassFlag.</p> <p>The values of ctxTable, ctxIdx and bypassFlag are derived as follows based on the entries for binIdx of the corresponding syntax element in Table 9-48:</p> <p>...</p> <ul style="list-style-type: none">– Otherwise, if the entry in Table 9-48 is equal to "bypass", the values of binIdx are decoded by invoking the DecodeBypass process as specified in clause 9.3.4.3.4 and the following applies:<ul style="list-style-type: none">– ctxTable is set equal to 0.– ctxIdx is set equal to 0.– bypassFlag is set equal to 1. <p>...</p> <p style="text-align: center;">Table 9-48 – Assignment of ctxInc to syntax elements with context coded bins</p> <table><tr><th rowspan="2">Syntax element</th><th colspan="6">binIdx</th></tr><tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>>= 5</th></tr><tr><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>sig_coeff_flag[][]</td><td>0..43 (clause 9.3.4.2.5)</td><td>na</td><td>na</td><td>na</td><td>na</td><td>na</td></tr><tr><td>coeff_abs_level_greater1_flag[]</td><td>0..23 (clause 9.3.4.2.6)</td><td>na</td><td>na</td><td>na</td><td>na</td><td>na</td></tr><tr><td>coeff_abs_level_greater2_flag[]</td><td>0..5 (clause 9.3.4.2.7)</td><td>na</td><td>na</td><td>na</td><td>na</td><td>na</td></tr><tr><td>coeff_abs_level_remaining[]</td><td>bypass</td><td>bypass</td><td>bypass</td><td>bypass</td><td>bypass</td><td>bypass</td></tr></table> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 225-227.</p>	Syntax element	binIdx						0	1	2	3	4	>= 5	...							sig_coeff_flag[][]	0..43 (clause 9.3.4.2.5)	na	na	na	na	na	coeff_abs_level_greater1_flag[]	0..23 (clause 9.3.4.2.6)	na	na	na	na	na	coeff_abs_level_greater2_flag[]	0..5 (clause 9.3.4.2.7)	na	na	na	na	na	coeff_abs_level_remaining[]	bypass	bypass	bypass	bypass	bypass	bypass
Syntax element	binIdx																																																
	0	1	2	3	4	>= 5																																											
...																																																	
sig_coeff_flag[][]	0..43 (clause 9.3.4.2.5)	na	na	na	na	na																																											
coeff_abs_level_greater1_flag[]	0..23 (clause 9.3.4.2.6)	na	na	na	na	na																																											
coeff_abs_level_greater2_flag[]	0..5 (clause 9.3.4.2.7)	na	na	na	na	na																																											
coeff_abs_level_remaining[]	bypass	bypass	bypass	bypass	bypass	bypass																																											

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<p>9.3.4.3 Arithmetic decoding process</p> <p>9.3.4.3.1 General</p> <p>Inputs to this process are ctxTable, ctxIdx and bypassFlag, as derived in clause 9.3.4.2, and the state variables ivlCurrRange and ivlOffset of the arithmetic decoding engine.</p> <p>Output of this process is the value of the bin.</p> <p>Figure 9-5 illustrates the whole arithmetic decoding process for a single bin. For decoding the value of a bin, the context index table ctxTable and the ctxIdx are passed to the arithmetic decoding process DecodeBin(ctxTable, ctxIdx), which is specified as follows:</p> <ul style="list-style-type: none"> – If bypassFlag is equal to 1, DecodeBypass() as specified in clause 9.3.4.3.4 is invoked. – Otherwise, if bypassFlag is equal to 0, ctxTable is equal to 0 and ctxIdx is equal to 0, DecodeTerminate() as specified in clause 9.3.4.3.5 is invoked. – Otherwise (bypassFlag is equal to 0 and ctxTable is not equal to 0), DecodeDecision() as specified in clause 9.3.4.3.2 is invoked. <p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 232.</p>

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<div><pre>graph TD; A([DecodeBin(ctxTable, ctxIdx, bypassFlag)]) --> B{bypassFlag == 1?}; B -- Yes --> C[DecodeBypass]; B -- No --> D{ctxTable == 0 && ctxIdx == 0?}; D -- Yes --> E[DecodeTerminate]; D -- No --> F[DecodeDecision(ctxTable, ctxIdx, bypassFlag)]; C --> G([Done]); E --> G; F --> G;</pre></div> <p>H.265v2(14)_F9-5</p>

Figure 9-5 – Overview of the arithmetic decoding process for a single bin (informative)

NOTE – Arithmetic coding is based on the principle of recursive interval subdivision. Given a probability estimation $p(0)$ and $p(1) = 1 - p(0)$ of a binary decision $(0, 1)$, an initially given code sub-interval with the range $ivlCurrRange$ will be subdivided into two sub-intervals having range $p(0) * ivlCurrRange$ and $ivlCurrRange - p(0) * ivlCurrRange$, respectively. Depending on the decision, which has been observed, the corresponding sub-interval will be chosen as the new code interval, and a binary code string pointing into that interval will represent the sequence of observed binary decisions. It is useful to distinguish between the most probable symbol (MPS) and the least probable symbol (LPS), so that binary decisions have to be identified as either MPS or LPS, rather than 0 or 1. Given this terminology, each context is specified by the probability $pLPS$ of the LPS and the value of MPS ($valMps$), which is either 0 or 1. The arithmetic core engine in this Specification has three distinct properties:

EXHIBIT 15
UNITED STATES PATENT NO. 9,036,701
CLAIM CHART FOR INFRINGEMENT OF CLAIM 1 BY HISENSE ACCUSED PRODUCTS

U.S. PATENT NO. 9,036,701	HISENSE ACCUSED PRODUCTS
	<ul style="list-style-type: none"> <li data-bbox="724 289 1963 451">– The probability estimation is performed by means of a finite-state machine with a table-based transition process between 64 different representative probability states $\{ pLPS(pStateIdx) \mid 0 \leq pStateIdx < 64 \}$ for the LPS probability $pLPS$. The numbering of the states is arranged in such a way that the probability state with index $pStateIdx = 0$ corresponds to an LPS probability value of 0.5, with decreasing LPS probability towards higher state indices. <li data-bbox="724 472 1963 602">– The range $ivlCurrRange$ representing the state of the coding engine is quantized to a small set $\{Q1, ..., Q4\}$ of pre-set quantization values prior to the calculation of the new interval range. Storing a table containing all 64×4 pre-computed product values of $Q_i * pLPS(pStateIdx)$ allows a multiplication-free approximation of the product $ivlCurrRange * pLPS(pStateIdx)$. <li data-bbox="724 623 1963 688">– For syntax elements or parts thereof for which an approximately uniform probability distribution is assumed to be given a separate simplified encoding and decoding bypass process is used. <p data-bbox="611 721 1488 753">ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 233.</p>